UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/722,614	11/26/2003	Ronald S. Karr	5760-09200	7091

35690 7590 07/17/2007

MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL, P.C.
P.O. BOX 398
AUSTIN, TX 78767-0398

EXAMINER	
TSAI, SHENG JEN	

ART UNIT	PAPER NUMBER
2186	

MAIL DATE	DELIVERY MODE
07/17/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/722,614
Filing Date: November 26, 2003
Appellant(s): KARR ET AL.

B. Noel Kivlin
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed April 26, 2007 appealing from the Office action mailed November 21, 2006.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Mater

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal Identifies the ground of rejections and the associated claims under rejection to be reviewed on appeal.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

2004/0030822	Rajan et al.	2-12-2004

(9) Grounds of Rejection

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1-28 are rejected under 35 U.S.C. 102(e) as being anticipated by Rajan et al. (U.S. Patent Application Publication 2004/0030822).

As to claim 1, Rajan et al. disclose **a storage subsystem** [Storage Virtualization by Layering Virtual Disk Objects on a File System (title); figure 1], **comprising**: **at least one storage device** [figure 1, 150 shows a plurality of storage disks]; **and a storage virtualization controller** [the corresponding storage virtualization controller is the multi-protocol storage appliance unit shown in figure 1, 100, comprising a processor (122), a memory (124) with storage operating system (200), a network adapter (125), a storage adapter (128) and a network target adapter (126); figure 2], **wherein the storage virtualization controller is communicatively coupled to the at least one storage device** [figure 1 shows that the communication between the controller (100) and the storage devices (150) is via the storage adapter (128)], **and wherein the storage virtualization controller is operable to**:

determining a metadata format [figure 4 shows the metadata] **usable to access data stored on the at least one storage device under a first operating system** [the operating system may be Windows (figure 1, 160a) or UNIX (figure 1, 160b)], **wherein the metadata format** [figure 4 shows the metadata] **is determined in response to a request by a host computer system** [the clients, figure 1, 160a or 160b] **to access the data, and wherein the metadata format is determined based on the host computer system** [the clients, figure 1, 160a or 160b] **running the first operating system** [First, the invention of Rajan et al. is directed toward a storage virtualization selection technique intended for a multi-protocol storage appliance to handle clients (i.e., host computers) running different types of operating systems (abstract). Figure 1 of Rajan et al. explicitly shows two clients, one running WINDOWS operating system (figure 1, 160a) and the other running UNIX operating system (figure 1, 160b); Second, since the host computers use different operating systems, different storage formats are needed to support different operating systems, and these differences are reflected in the corresponding operating system metadata, as explicitly disclosed by Rajan et al. "For example, a client 160a running the Windows operating system may communicate with the storage appliance 100 using the Common Internet File System (CIFS) protocol over TCP/IP. On the other hand, a client 160b running the UNIX operating system may communicate with the multi-protocol appliance using either the Network File System (NFS) protocol over TCP/IP or the Direct Access File System (DAFS) protocol over a virtual interface (VI) transport in accordance with a remote DMA (RDMA) protocol over TCP/IP. It will be apparent to those skilled in the art that other

clients running other types of operating systems may also communicate with the integrated multi-protocol storage appliance using other file access protocols (paragraph 0026)"; Third, figure 2 of Rajan et al. shows, as the Appellants observe, that both CIFS (for WINDOW operating system, see above) and NFS (for UNIX operating system, see above) are installed and made available during the installation phase. However, the determination as to which one of the CIFS and NFS should be used is made dynamically depending on which client (figure 1, 160a or 160b) initiates the access request "The storage adapter 128 cooperates with the storage operating system 200 executing on the storage appliance to access information requested by the clients (paragraph 0030)". In other words, the determination of which one to use is made dynamically in response to a request by a particular client (figure 1, 160a or 160b) to access the data; figures 4 and 5 show examples of the metadata].

generate operating system metadata in accordance with the determined metadata format [The storage adapter 128 cooperates with the storage operating system 200 executing on the storage appliance to access information requested by the clients (paragraph 0030); For example, a client 160a running the Windows operating system may communicate with the storage appliance 100 using the Common Internet File System (CIFS) protocol over TCP/IP. On the other hand, a client 160b running the UNIX operating system may communicate with the multi-protocol appliance using either the Network File System (NFS) protocol over TCP/IP or the Direct Access File System (DAFS) protocol over a virtual interface (VI) transport in accordance with a remote DMA (RDMA) protocol over TCP/IP. It will be apparent to those skilled in the

art that other clients running other types of operating systems may also communicate with the integrated multi-protocol storage appliance using other file access protocols (paragraph 0026); since the host computers use different operating systems with different storage formats, these differences are reflected in the corresponding operating system metadata; figure 4, 410 shows the metadata generated, including type (412), size (414), time stamps (416), UID (418), GID (420) and Xinode (430); figure 5, 501, 512, 522, 542, 552 and 562 provide more examples of metadata] **for the at least one storage device** [figure 1, 150 shows a plurality of storage disks], **wherein the operating system metadata emulates a storage volume hosted under the first operating system** [figure 3 shows the storage volume information indicated by VDISK module (330) including LUN (Logical Unit Number), IGROUP and Map Binding; as used herein, the term "storage operating system" generally refers to the computer-executable code operable on a computer that manages data access and may, in the case of a multi-protocol storage appliance, implement data access semantics, such as the Data ONTAP storage operating system, which is implemented as a microkernel. The storage operating system can also be implemented as an application program operating over a general-purpose operating system, such as UNIX or Windows NT, or as a general-purpose operating system with configurable functionality, which is configured for storage applications as described herein (paragraph 0035)]; **and send the operating system metadata to the host computer system** [the corresponding host computer system is the client systems shown in figure 1, 160a and 160b], **wherein the operating system metadata enables the host computer system**

to recognize the storage device as the storage volume hosted under the first operating system [Whereas clients of a NAS-based network environment have a storage viewpoint of files, the clients of a SAN-based network environment have a storage viewpoint of blocks or disks. To that end, the multi-protocol storage appliance 100 presents (exports) disks to SAN clients through the creation of logical unit numbers (luns) or vdisk objects. A vdisk object (hereinafter "vdisk") is a special file type that is implemented by the virtualization system and translated into an emulated disk as viewed by the SAN clients. The multi-protocol storage appliance thereafter makes these emulated disks accessible to the SAN clients through controlled exports, as described further herein (paragraph 0023)].

As to claim 2, Rajan et al. teach that **the operating system metadata enables a block storage I/O stack in the first operating system on the host computer system to recognize the storage device as a partition** [To that end, the multi-protocol storage appliance 100 presents (exports) disks to SAN clients through the creation of logical unit numbers (luns) or vdisk objects. A vdisk object (hereinafter "vdisk") is a special file type that is implemented by the virtualization system and translated into an emulated disk as viewed by the SAN clients. The multi-protocol storage appliance thereafter makes these emulated disks accessible to the SAN clients through controlled exports, as described further herein (paragraph 0023). Note that logical unit numbers (luns) and vdisk objects are both special forms of partition; figure 3 shows that data being partitioned into the form of "record" (372)].

As to claim 3, Rajan et al. teach that **the operating system metadata enables a block storage I/O stack in the first operating system on the host computer system to recognize the storage device as a host-virtual object** [To that end, the multi-protocol storage appliance 100 presents (exports) disks to SAN clients through the creation of logical unit numbers (luns) or vdisk objects. A vdisk object (hereinafter "vdisk") is a special file type that is implemented by the virtualization system and translated into an emulated disk as viewed by the SAN clients. The multi-protocol storage appliance thereafter makes these emulated disks accessible to the SAN clients through controlled exports, as described further herein (paragraph 0023). Note that data is viewed by the clients as virtual disk (vdisk) object].

As to claim 4, Rajan et al. teach that **the operating system metadata enables a driver on the host computer system to recognize the storage device as an enclosed volume** [To that end, the multi-protocol storage appliance 100 presents (exports) disks to SAN clients through the creation of logical unit numbers (luns) or vdisk objects], **wherein the driver is layered above a block storage I/O stack in the first operating system** [The storage operating system comprises a series of software layers organized to form an integrated network protocol stack ... (paragraph 0037)].

As to claim 5, Rajan et al. teach that **the storage virtualization controller is operable to configure the operating system metadata in response to a requirement of the first operating system** [The vdisk is thereafter created as a storage object within a volume and, thus, inherits the underlying reliability configuration associated with that volume (abstract); The file server, or filer, may be further

configured to operate according to a client/server model of information delivery to thereby allow many client systems (clients) to access shared resources, such as files, stored on the filer (paragraph 0003)].

As to claim 6, Rajan et al. teach that **a management environment is configured to supply operating system types and operating system metadata configuration requirements to the storage virtualization controller** [The file server, or filer, may be further configured to operate according to a client/server model of information delivery to thereby allow many client systems (clients) to access shared resources, such as files, stored on the filer (paragraph 0003)], **wherein the operating system types comprise the first operating system** [The storage operating system can also be implemented as an application program operating over a general-purpose operating system, such as UNIX or Windows NT, or as a general-purpose operating system with configurable functionality, which is configured for storage applications as described herein (paragraph 0035); figure 1 shows that client 160a runs WINDOWS operating system and client 160b runs UNIX operating system].

As to claim 7, Rajan et al. teach that **in generating the operating system metadata for the storage device, the storage virtualization controller is operable to add a storage property to identify an offset and a length of the storage volume** [figure 4, 410 shows the metadata generated, including type (412), size (414), time stamps (416), UID (418), GID (420) and Xinode (430); figure 4 of Rajan et al. shows the format of the operating system metadata and the associated data (paragraph 0044). The metadata comprises a plurality of fields describing the storage property,

including the "size" (414) and the "xinode" (430). The "size" field provides information regarding the size (or length) of a storage element (e.g. a file) while the "xinode" field containing a "pointer" (i.e., the corresponding offset) that references another inode structure (paragraph 0044); figure 5 Rajan et al. further illustrates the detailed structure of the storage volume using the property of "size" and "xinode" contained in the metadata; figure 3 shows the storage volume information indicated by VDISK module (330) including LUN (Logical Unit Number), IGROUP and Map Binding].

As to claim 8, Rajan et al. teach that **an operation is provided to configure operating system types and operating system metadata configuration requirements for generating the operating system metadata, wherein the operating system types comprise the first operating system** [The vdisk is thereafter created as a storage object within a volume and, thus, inherits the underlying reliability configuration associated with that volume (abstract); The file server, or filer, may be further configured to operate according to a client/server model of information delivery to thereby allow many client systems (clients) to access shared resources, such as files, stored on the filer (paragraph 0003)].

As to claim 9, Rajan et al. teach that **the storage virtualization controller is operable to receive user input to select one of a plurality of operating system types for the operating system metadata, wherein the operating system types comprise the first operating system** [The file server, or filer, may be further configured to operate according to a client/server model of information delivery to

thereby allow many client systems (clients) to access shared resources, such as files, stored on the filer (paragraph 0003)].

As to claim 10, Rajan et al. teach that **the storage virtualization controller is operable to send an operating system metadata configuration instruction to the storage device through a vendor-unique I/O request to the storage device** [Whereas clients of a NAS-based network environment have a storage viewpoint of files, the clients of a SAN-based network environment have a storage viewpoint of blocks or disks ; To that end, the multi-protocol storage appliance 100 presents (exports) disks to SAN clients through the creation of logical unit numbers (luns) or vdisk objects. A vdisk object (hereinafter "vdisk") is a special file type that is implemented by the virtualization system and translated into an emulated disk as viewed by the SAN clients. The multi-protocol storage appliance thereafter makes these emulated disks accessible to the SAN clients through controlled exports, as described further herein (paragraph 0023); figure 4, 410 shows the metadata generated, including type (412), size (414), time stamps (416), UID (418), GID (420) and Xinode (430)].

As to claim 11, Rajan et al. teach that **the operating system metadata emulates a storage volume hosted under a first operating system and one or more additional operating systems** [the term "storage operating system" generally refers to the computer-executable code operable on a computer that manages data access and may, in the case of a multi-protocol storage appliance, implement data access semantics, such as the Data ONTAP storage operating system, which is

implemented as a microkernel. The storage operating system can also be implemented as an application program operating over a general-purpose operating system, such as UNIX or Windows NT, or as a general-purpose operating system with configurable functionality, which is configured for storage applications as described herein (paragraph 0035); figure 1 shows that client 160a runs WINDOWS operating system and client 160b runs UNIX operating system]; **and wherein the operating system metadata enables a layered driver on the host computer system to recognize the storage device** [The storage operating system comprises a series of software layers organized to form an integrated network protocol stack ... (paragraph 0037)].

As to claim 12, Rajan et al. teach **using a layered driver on the host computer system to provide access to a storage volume mapped within a Logical Unit, wherein the Logical Unit is provided by an external device or an external virtualization layer** [figure 3 shows the storage volume information indicated by VDISK module (330) including LUN (Logical Unit Number), IGROUP and Map Binding; To that end, the multi-protocol storage appliance 100 presents (exports) disks to SAN clients through the creation of logical unit numbers (luns) or vdisk objects (paragraph 0023); The storage operating system comprises a series of software layers organized to form an integrated network protocol stack ... (paragraph 0037)].

As to claim 13, Rajan et al. teach that **a management environment is configured to supply a preferred name of the storage device to software on the host computer system** [a storage operating system 200 that provides a virtualization

system (and, in particular, a file system) to logically organize the information as a hierarchical structure of named directory, file and virtual disk (vdisk) storage objects on the disks 130 (paragraph 0022)].

As to claim 14, refer to "As to claim 1" presented earlier in this Office Action.

As to claim 15, refer to "As to claim 2" presented earlier in this Office Action.

As to claim 16, refer to "As to claim 3" presented earlier in this Office Action.

As to claim 17, refer to "As to claim 4" presented earlier in this Office Action.

As to claim 18, refer to "As to claim 5" presented earlier in this Office Action.

As to claim 19, refer to "As to claim 6" presented earlier in this Office Action.

As to claim 20, refer to "As to claim 7" presented earlier in this Office Action.

As to claim 21, refer to "As to claim 8" presented earlier in this Office Action.

As to claim 22, refer to "As to claim 9" presented earlier in this Office Action.

As to claim 23, refer to "As to claim 10" presented earlier in this Office Action.

As to claim 24, refer to "As to claim 11" presented earlier in this Office Action.

As to claim 25, refer to "As to claim 12" presented earlier in this Office Action.

As to claim 26, refer to "As to claim 13" presented earlier in this Office Action.

As to claim 27, refer to "As to claim 1" presented earlier in this Office Action. Further, figure 6 of Rajan et al. shows the flowchart of the computer programs that implement the storage operating system.

As to claim 28, refer to "As to claim 1" presented earlier in this Office Action.

(10) Response to Arguments

Appellants' arguments have been fully and carefully considered with Examiner's answers set forth below.

Answer to Arguments on Claims 1-6, 8-19 and 21-28

Appellants contend that the reference (Rajan et al., US Patent Application Publication 2004/0030822) does not teach the limitation of "determine a metadata format usable to access data stored on the at least one storage device under a first operating system, wherein the metadata format is determined in response to a request by a host computer system to access the data," because, in the invention of Rajan et al., the metadata format for the data on the storage appliance is fixed according to the requirements of the storage operating system installed on the storage appliance, and the metadata format is determined when the storage operating system is installed before any request by a host computer system to access the data instead of being determined in response to a request by a host computer system to access the data. The Examiner disagrees with this assessment due to the following reasons:

First, the invention of Rajan et al. is directed toward a storage virtualization selection technique intended for a multi-protocol storage appliance to handle clients (i.e., host computers) running different types of operating systems [abstract]. Figure 1 of Rajan et al. explicitly shows two types of clients, one running WINDOWS operating system [figure 1, 160a] and the other running UNIX operating system [figure 1, 160b].

Second, since the host computers (i.e. clients) use different operating systems, different storage formats are needed to support different operating systems, and these

differences are reflected in the corresponding operating system metadata, as explicitly disclosed by Rajan et al. [For example, a client 160a running the Windows operating system may communicate with the storage appliance 100 using the Common Internet File System (CIFS) protocol over TCP/IP. On the other hand, a client 160b running the UNIX operating system may communicate with the multi-protocol appliance using either the Network File System (NFS) protocol over TCP/IP or the Direct Access File System (DAFS) protocol over a virtual interface (VI) transport in accordance with a remote DMA (RDMA) protocol over TCP/IP. It will be apparent to those skilled in the art that other clients running other types of operating systems may also communicate with the integrated multi-protocol storage appliance using other file access protocols (paragraph 0026)].

Third, figure 2 of Rajan et al. shows that both CIFS (figure 2, 222, for WINDOW operating system, see above) and NFS (figure 2, 220, for UNIX operating system, see above) are installed and made available during the installation phase. However, the determination as to which one of the CIFS and NFS should be used is made dynamically depending on which client (figure 1, 160a or 160b) initiates the access request [The storage adapter 128 cooperates with the storage operating system 200 executing on the storage appliance to access information requested by the clients (paragraph 0030)]. In other words, although the installation of both CIFS and NFS is fixed, the determination of which one to use is made dynamically in response to a request by a client to access the data.

Therefore, the Examiner's position regarding the patentability of claims 1-6, 8-19 and 21-28 remains the same as stated in the previous Office Action.

Answer to Arguments on Claims 7 and 20

Appellants contend that claims 7 and 20 are patentable over Rajan et al. in that Rajan et al. fail to teach the limitation of "in generating the operating system metadata for the storage device, the storage virtualization controller is operable to add a storage property to identify an offset and a length of the storage volume,". Appellants contend that the metadata, disclosed by Rajan et al. in figure 4 does not correspond to a storage property identifying an offset and a length of a storage volume. The Examiner disagrees.

Figure 4 of Rajan et al. shows the format of the operating system metadata and the associated data [paragraph 0044]. The metadata comprises a plurality of fields describing the storage property, including the "size" (414) and the "xinode" (430). The "size" field provides information regarding the size (or length) of a storage element (e.g. a file) while the "xinode" field containing a "pointer" (i.e., the corresponding offset) that references another inode structure [paragraph 0044]. Figure 5 of Rajan et al. further illustrates the detailed structure of the storage volume using the property of "size" and "xinode" contained in the metadata.

Thus, Rajan et al. clearly teach this particular limitation.

Therefore, the Examiner's position regarding the patentability of claims 7 and 20 remains the same as indicated in the previous Office Action.

(11) Related Proceedings Appendix

None.

Sheng-Jen Tsai Examiner Art Unit 2186 Sheng-Jen Tsai 7/5/2007
Matthew Kim Supervisory Patent Examiner Art Unit 2186 /Matthew M. Kim/
Lynne H Browne Appeal Practice Specialist, TQAS Technology Center 2100 /Lynne H Browne/

July 5, 2007